REMARKS/ARGUMENTS

Applicant submits this Amendment and Response to respond to the Office Action dated January 9, 2006. Claims 1, 15, and 22 have been amended without intending to abandon or to dedicate to the public any patentable subject matter. Accordingly, Claims 1-3, 5-16, and 18-30 are now pending.

Claims 1-29 have been rejected under 35 U.S.C. §103(a) as being unpatentable over U.S. Patent No. 6,104,721 to Hsu ("Hsu") in view of either U.S. Patent No. 6,862,623 to Odhner et al. ("Odhner"), U.S. Patent No. 6,314,447 to Lea et al. ("Lea"), or U.S. Patent No. 6,484,265 to Borkar et al. ("Borkar"). In order for a rejection under 35 U.S.C. §103 to be proper, there must be some suggestion or motivation to modify the reference or to combine the reference teachings, there must be a reasonable expectation of success, and the prior art references relied upon must teach, suggest, or describe each and every element as set forth in the claims. (MPEP §2143.) However, all of the limitations set forth in the pending claims are not taught, suggested, or described by any of the above mentioned references. Accordingly, reconsideration and withdrawal of the rejections of the claims are respectfully requested.

Hsu, the primary reference, is directed to a dynamic resource allocation multiprocessor communications board. This board has several identical processors that can be coupled to one another in series. When a task needs to be processed, the controller determines if a single processor has the capability to perform the task. If the controller determines the single processor does have the required capabilities, the task is assigned to the processor. If the task requires more processing power than a single processor has, the controller selects at least two of the processors and forms a pipeline processing combination by enabling at least one of the serial ports to couple the at least two selected processors in series. (Hsu col. 3, ll. 41-52.) Hsu therefore meets the processing demands of a given task that the processing power a single processor is unable to meet by linking at least two of the processors in the processing bank to form a pipeline processing combination. (Hsu, col. 3, ll. 45-51.) The resource usage status field 178 in Hsu stores "Data representative of the status of the available processing power of the corresponding processor." (Hsu, col. 9, ll. 61-62.) The resource usage status field only tracks the amount of

resources that are assigned to a given processor. There is no field for storing the maximum resource capabilities of the processors in Hsu. Furthermore, there is no teaching or suggestion to provide such a field, because the maximum resource capabilities of each processor are the same.

Odhner is directed toward a system and method for capacity planning of server resources. Measured data is used along with estimations and extrapolation to provide reportedly accurate capacity planning results. A user "inputs a load desired to be handled by the server cluster and receives a recommendation for server cluster changes that will enable the server cluster to adequately handle the load." (Odhner, col. 2, ll. 58-61.) The recommended server cluster changes typically involve adding more servers to the server cluster. A theoretical maximum load value for a server cluster is obtained using a "pre-defined load table that contains empirically-derived maximum load values handled by servers having a known amount of memory." (Odhner, col. 2, ll. 29-31.) The utilization of the table in Odhner is vastly different from the way the table is utilized in the present invention. Namely, Odhner uses a table to determine a maximum load (number of requests per second) that can be handled by a given server cluster. Odhner does not teach using the table to determine what computer resource a particular task should be assigned to.

Lea is directed toward a system that determines processing capabilities of devices in an electronic network. A host device monitors the network to determine whether a system user has recently connected a new device to the electronic network. The host performs a discovery process by querying relevant configuration information stored in the self-describing data of the new device. Then based on the current load levels of the existing processors, the host re-allocates tasks among the processors and the new device. This load balancing ensures that no device is too heavily loaded. A device application determines whether a hosted device is capable of processing a given task by referencing the self-describing data stored in the hosted device. If a particular hosted device cannot handle a given task, then additional hosted devices are assigned to the task in order to balance the load across multiple processors. The self-describing data of each hosted device is stored in the hosted device to be referenced by other devices as it is required. Lea does not teach, suggest, or describe using a table in order to determine whether a hosted device has sufficient processor capabilities to perform the required processing task.

Borkar is generally directed toward a system having a processor and control circuitry. The control circuitry controls a body bias signal in order to control parameters of the processor. Control circuitry may further control settings of voltage supplied to the processor and a clock signal transmitted to the processor. The Borkar reference does not teach, suggest, or describe the use of a table in connection with assigning tasks to computer resources.

As noted above there is no discussion in any of the above cited references of using a table to determine whether or not a task should be assigned to a resource or how a task should be assigned to a set of resources. More particularly, none of the above-noted references disclose at least the following italicized portions of independent Claim 1:

> 1. A method for dynamically allocating tasks in a computer system, comprising:
>
> assigning a maximum computer resource load to each of a plurality of computer platforms, *wherein a first of said computer platforms has a first maximum computer resource capability and load and a second of said computer platforms has a second maximum resource capability and load*;
>
> *providing a computer resources table, wherein indications of maximum computer resource capabilities for each of said plurality of computer platforms are maintained*;
>
> assigning a computer resource requirement to a task;
>
> *referencing said computer resources table for maximum computer resource capabilities of at least one of said plurality of computer platforms*;
>
> *assigning said task to a selected one of said plurality of computer platforms, wherein said task is assigned to said selected computer platform based on said resource requirement of said task and said maximum resource load of said selected platform*; and
>
> performing said task in connection with said selected computer platform.

As highlighted above, all of the cited references fail to teach, suggest, or describe providing a computer resources table that has indications of maximum computer resource capabilities for each of the computer platforms. Moreover, none of the above cited references teach referencing such a computer resources table for maximum computer resource capabilities and then assigning a task to one of the computer platforms. Furthermore, the processors described in Hsu have the same maximum processing capabilities and therefore indications of

maximum computer resource capabilities are not stored in a table. Therefore, for at least these reasons, Claim 1 and dependent Claims 2, 3, and 5-14 are not obvious in view of any of the cited references, and the rejections of Claim 1 and dependent claims therefrom should be reconsidered and withdrawn.

The above noted references also fail to disclose at least the following italicized portions of independent Claim 15:

> 15. A method for dynamically allocating computer processor tasks, comprising:
>
> dynamically specifying a first maximum capability of a first computer processor;
>
> dynamically specifying a second maximum capability of a second computer processor;
>
> *maintaining a computer processor capability table, wherein dynamically adjusted capability values for said first and second computer processors are stored that are related to said first and second maximum capabilities of said first and second computer processors*;
>
> receiving a first task requiring processing, wherein a first processor load value is associated with said first task;
>
> *referencing said computer processor capability table to determine that said first processor load value of said first task is greater than said first capability of said second computer processor;*
>
> *assigning said first task to said first computer processor, wherein said first processor load value is less than said first capability of said first computer processor*; and
>
> processing said first task using said first computer processor.

As noted above, none of the cited references teach, suggest, or describe maintaining a computer processor capability table where dynamically adjusted capability values for first and second computer processors are stored. Moreover, none of the cited references teach referencing the table to determine whether the first processor load value of the first task is greater than the capability of the second computer processor. Furthermore, the cited references fail to teach assigning the first task to the first computer processor where the processor load of the task is less than the capability of the first computer processor. Therefore, for at least these reasons, Claim 15

and dependent Claims 16 and 18-21 are not obvious in view of any of the cited references, and the rejections of Claim 15 and dependent claims therefrom should be reconsidered and withdrawn.

Additionally, the above noted references fail to teach, suggest, or describe at least the following italicized portions of independent Claim 22:

> 22. A computer resource allocation system, comprising:
> at least a first computer platform comprising at least a first computer resource and a second computer platform comprising at least a second computer resource, *wherein said at least a first computer platform has a first task type capability and a first resource amount capability, wherein said second computer platform has a second task type capability and a second resource amount capability, wherein said first and second task type capabilities do not have to be the same, and wherein said first and second resource amount capabilities do not have to be the same*;
> processing software running on a server processor, comprising:
> *memory including a table, wherein said first and second resource amount capabilities of said first and second computer platforms are stored in said table*; and
> *a software task allocation unit, operable to reference resource amount capabilities in said table and further operable to allocate a task to a selected one of said first or second computer platforms based on said task type capability and said referenced resource amount capability*, and wherein said task is completed in connection with said selected one of said first or second computer platforms having a task type capability required to complete said task and a resource amount capability sufficient to complete said task.

Again, the cited references fail to teach, suggest, or describe a system that comprises a memory including a table for storing first and second resource amount capabilities of first and second computer platforms. Additionally, the cited references do not suggest a software task allocation unit that is operable to reference the resource amount capabilities in the table and is further operable to allocate a task to one of the first or second platforms based on the referenced resource amount capability. Therefore, for at least these reasons, Claim 22 and dependent Claims 23-29 are not obvious in view of any of the cited references, and the rejections of Claim 22 and the dependent claims therefrom should be reconsidered and withdrawn.

Application No. 10/002,073

Furthermore, the combination of Hsu and Odhner would essentially result in a determination of the maximum number of loads the processors in Hsu would be able to process. The combination of Odhner and Hsu does not fairly teach, suggest, or describe allocating computer processor tasks. Indeed, if one of skill in the art were to combine the teachings of Odhner and Hsu the result would simply be a recommendation of how to connect various processors in Hsu (which all have the same processing capabilities) or how many more processors should be connected to the pipeline described in Hsu.

Additionally, Applicant notes that the Examiner has failed to reject Claim 30 under any of the above-noted art. Therefore, the Applicant can only assume that the Examiner has found Claim 30 allowable in view of the cited art.

Based upon the foregoing, Applicant believes that all pending claims are in condition for allowance and such disposition is respectfully requested. In the event that a telephone conversation would further prosecution and/or expedite allowance, the Examiner is invited to contact the undersigned.

Respectfully submitted,

SHERIDAN ROSS P.C.

By: [signature]

Matthew R. Ellsworth
Registration No. 56,345
1560 Broadway, Suite 1200
Denver, Colorado 80202-5141
(303) 863-9700

Date: April 6, 2006